UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/791,602	03/02/2004	Carl A. Waldspurger	A043	2029

36378 7590 10/26/2010

VMWARE, INC.
DARRYL SMITH
3401 Hillview Ave.
PALO ALTO, CA 94304

EXAMINER	
TURCHEN, JAMES R	

ART UNIT	PAPER NUMBER
2439	

NOTIFICATION DATE	DELIVERY MODE
10/26/2010	ELECTRONIC

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Notice of the Office communication was sent electronically on above-indicated "Notification Date" to the following e-mail address(es):

ipteam@vmware.com
ipadmin@vmware.com

Office Action Summary	Application No.	Applicant(s)	
	10/791,602	WALDSPURGER ET AL.	
	Examiner	Art Unit	
	JAMES TURCHEN	2439	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE *3* MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *03 June 2010*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *2-4,6-19,21-33 and 35-62* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *2-4,6-19,21-33 and 35-62* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

Continued Examination Under 37 CFR 1.114

A request for continued examination under 37 CFR 1.114, including the fee set forth in 37 CFR 1.17(e), was filed in this application after final rejection. Since this application is eligible for continued examination under 37 CFR 1.114, and the fee set forth in 37 CFR 1.17(e) has been timely paid, the finality of the previous Office action has been withdrawn pursuant to 37 CFR 1.114. Applicant's submission filed on 06/03/2010 has been entered.

Response to Arguments

Applicant's arguments filed 06/03/2010 have been fully considered but they are not persuasive. Examiner respectfully disagrees with applicant that "ensuring that the program is not executed without performing the verifying" distinguishes over the Das reference. Das discloses that an instruction is sent to a fetch and decode unit [*figure 5, col 6, 16-18, first functional units can be fetch and decode unit*], then it is determined whether the instruction is associated with a virus [*col 6, 22-25*] and proceeds to send the instruction to be executed to the second functional unit [*col 6, 26-35, dispatch and execution unit*]. The instruction is not yet executed and it is scanned prior to being executed. When a virus is detected, the instruction is flushed (not executed) [*column 8 lines 3-23*]. Therefore, Das ensures that the program is not executed without performing the verifying (virus check). The program is dynamically verified and executed thereafter. The virus scan is considered by the examiner to be an interruption in the execution of the program.

Claim Rejections - 35 USC § 101

35 U.S.C. 101 reads as follows:

> Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title.

Claims 35-44 and 46-49 are rejected under 35 U.S.C. 101 because the claimed invention is directed to non-statutory subject matter.

The broadest reasonable interpretation of a claim drawn to a computer readable medium (also called machine readable medium and other such variations) typically covers forms of non-transitory tangible media and transitory propagating signals *per se* in view of the ordinary and customary meaning of computer readable media, **particularly when the specification is silent. See MPEP 2111.01** When the broadest reasonable interpretation of a claim covers a signal *per se*, the claim must be rejected under U.S.C. 101 as covering non-statutory subject matter. See In re Nuijten, 500 F.3d 1346, 1356-56 (Fed Cir. 2007)(transitory embodiments are not directed to statutory subject matter)

The USPTO suggests the following approach to overcome this 101 rejection. A claim drawn to such a computer readable medium that covers both transitory and non-transitory may be amended to marrow the claim to cover only statutory embodiments to avoid a rejection under 35 U.S.C. 101 by adding the limitation "non-transitory" to the claim.

Claim Rejections - 35 USC § 102

The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (e) the invention was described in (1) an application for patent, published under section 122(b), by another filed in the United States before the invention by the applicant for patent or (2) a patent granted on an application for patent by another filed in the United States before the invention by the applicant for patent, except that an international application filed under the treaty defined in section 351(a) shall have the effects for purposes of this subsection of an application filed in the United States only if the international application designated the United States and was published under Article 21(2) of such treaty in the English language.

Claims 2-4, 35-37, 50-52 and 62 are rejected under 35 U.S.C. 102(e) as being anticipated by Das.

Regarding claims 2, 35 and 50:

Das discloses a method for protecting a computer from unauthorized code, the computer including at least one processor that executes instructions stored in a memory, the memory being organized into separately addressable memory blocks, the method comprising:

executing a program, the program having a series of computer-executable instructions [*column 3 lines 55-67 and column 5 lines 16-26, software programs and instructions*];

verifying that the program is valid, the program being valid when the program does not include the unauthorized code [*figure 6, item 607 and column 6 line 65-column 7 line 42, is instruction associated with a virus?*];

ensuring that the program is not executed without dynamically performing the verifying, the verifying based on a next instruction to be executed in the series of computer-executable instructions in the program [*column 6 lines 45-64, the instruction is fetched/decoded but not yet executed; column 8 lines 3-23, if there is a virus, the processor pipeline is flushed and drops the instruction associated with a virus*]; and

continuing execution of the program after dynamically performing the verifying by executing the next instruction as long as the verifying determines that the program is valid and generating a protective response when the verifying determines that the program is not valid [*column 7 lines 28-42, executing in parallel for multiple instructions, as the next instruction is passed to be executed another instruction is fetched and decoded*];

wherein the verifying that the program is valid comprises:

identifying a the next instruction of the series of computer executable instructions to be executed in the program [*column 6 line 65-column 7 line 42*];

for the next instruction to be executed in the program, determining an identifying value for a memory block that contains the next instruction [*column 8 lines 29-42*];

determining whether the identifying value satisfies a validation condition, wherein the determining as to whether the identifying value satisfies the validation condition requires comparing the identifying value of the memory block with a set of reference values [*column 8 lines 28-42*]; and

determining that the program is valid when the validation condition is satisfied [*column 7 lines 1-6 and column 8 lines 28-42*].

Regarding claims 3, 36 and 51:

Das discloses the method of claim 2, wherein the validation condition is that the identifying value of the memory block matches any reference value in the set of reference values [*figure 8 and column 8 lines 29-42*].

Regarding claims 4, 37 and 52:

Das discloses the method of claim 2, wherein the validation condition is that the identifying value of the memory block differs from each reference value in the set of reference values [*figure 8 and column 8 lines 29-42*].

Regarding claim 62:

Das discloses the method of claim 2, wherein ensuring that the program is not executed without dynamically performing the verifying comprises interrupting execution of the series of computer-executable instructions of the program while dynamically performing the verifying [*column 7 lines 1-42, the scanning in between the fetch/decode and execute is considered a small interruption*].

Claim Rejections - 35 USC § 103

The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

Claims 6, 7, 12, 13, 14, 15, 17, 19, 21, 26, 27, 30, 39, 48, 49, 55, 60, and 61 are rejected under 35 U.S.C. 103(a) as being unpatentable over Das in view of Nachenberger '013.

Das discloses a method for verifying computer instructions in a computer that includes at least one processor that executes instructions stored in memory, the memory being organized into separately addressable memory blocks, the method comprising:

executing a program, the program having a series of computer-executable instructions [*column 3 lines 55-67 and column 5 lines 16-26, software programs and instructions*];

verifying that the program is valid, the program being valid when the program does not include the unauthorized code, the verifying based on a current instruction to be executed when executing the series of instructions in the program [*figure 6, item 607 and column 6 line 65-column 7 line 42, is instruction associated with a virus?*];

generating a protective response when the verifying determines that the program is not valid [*column 8 lines 28-42*];

wherein the verifying that the program is valid comprises:

identifying a current instruction to be executed when executing the series of instructions, the current instruction being one of the series of instructions being executed identified for submission to the processor for execution and not yet executed at a time of the identifying [*column 6 line 65-column 7 line 42*];

for at least the current instruction, computing a hash value as a function of a subset of contents of a current memory block that contains the current instruction [*column 8 lines 28-42, a "virus signature is a unique string of bits, like a fingerprint, that it can be used to detect and identify specific viruses" (taken from Webopedia.com)*];

determining, during the executing of the series of instructions, whether the hash value satisfies a validation condition by comparing the hash value of the current memory block with a set of reference values [*column 8 lines 28-42*]; and

determining that the program is valid when the hash value satisfies the validation condition, and determining that the program is not valid when the hash value does not satisfy the validation condition [*column 8 lines 28-42*];

Das does not disclose:

wherein the computing of the hash value comprises applying a mask to the current memory block, the mask being a data structure that designates at least one byte of the current memory block to be ignored in the computing of the hash value, the data structure designating less than an entire memory block so that the hash value is based on only part of the contents of the current memory block.

Nachenberg '013 discloses page locations are accessed and scanned a byte at a time to determine whether each byte is a selected byte. If not, the next byte is scanned and tested until it is determined that no more bytes remain [*column 12 lines 20-43, the selected page locations are scanned and the non-selected ones are ignored*]. It would have been obvious to one of ordinary skill in the art at the time of invention to combine the method of Das with the method of Nachenberg '013 as claimed by known methods with no change in their respective functions, and the combination would have yielded predictable results to one of ordinary skill in the art at the time of invention.

Regarding claims 7 and 55:

Das and Nachenberg '013 disclose the method of claim 6, further comprising:

identifying, an indeterminate portion of the current memory block, the indeterminate portion being non-indicative of validity of the current memory block as a whole [*013, figure 5, checks for register modifications*]; and

configuring the mask so that the mask designates at least the indeterminate portion to be ignored when generating the hash value [*013, figure 5, 512, exclude all viruses that cannot perform memory write with non-initialized index register*].

Regarding claim 12:

013, 510 and Das disclose the method of claim 8, the determining of the identifying value for the memory block and the determining of whether the identifying value satisfies the validation condition are performed only when the structure does not indicate that the memory block is valid, the method further comprising:

modifying the structure so that the memory block is not indicated as being valid when the identifying value satisfies the validation condition [*013, it is inherent to mark the file as virus-free or contaminated*].

Regarding claim 13:

Das and Nachenberg '013 disclose the method of claim 12, further comprising:

sensing modification of one of the memory blocks that the structure indicates is valid and, in response to the modification, setting its indication in the structure to indicate that the memory block is not valid [*013; figure 5, 502-512*].

Regarding claims 14 and 60:

Das and Nachenberg '013 disclose the method of claim 8, but does not disclose the method further comprising:

determining a branch history for the next instruction; and

checking whether the memory blocks in which instructions in the branch history are located are valid, the validation condition including the requirement that each checked memory block in the branch history is valid.

Examiner takes official notice that branch prediction is well known in the art at the time of invention. All the claimed elements were known and it would have been obvious to one or ordinary skill in the art to combine the elements by known methods with no change in their respective functions, and the combination would have yielded predictable results to one of ordinary skill in the art at the time of invention.

Regarding claims 15 and 61:

Das and Nachenberg '013 disclose the method of claim 2, wherein the determining of the identifying value and the determining as to whether the validation condition has been satisfied are performed only after a triggering event occurs [*it is inherent that a file be checked after the occurrence of a triggering event whether by a user, a time-based trigger, an active trigger, etc.*].

Regarding claim 17:

Das and Nachenberg '013 disclose the method of claim 15, but does not disclose in which the triggering event is an attempted execution of any instruction located on any unverified memory block. Examiner takes official notice that dynamically scanning a program as it is being opened or executed is known. It would have been obvious to one of ordinary skill in the art at the time of invention to modify the method of Nachenberg to scan a program in order to ensure the integrity of an executable program while it is

running or to detect infections between a programs integrity check and execution [A Generic Virus Scanner in C++, page 6, section 2.5]

Regarding claim 19:

Das and Nachenberg '013 disclose the method of claim 15, further comprising triggering the verification of the computer instructions depending on an identity of a user of the computer, the user having caused the next instruction to be identified for execution [*it is inherent that a computer OS has a method for access control based on the user of the system in that the program will not load if the user does not have permission to run it*].

Regarding claim 21:

Das and Nachenberg '013 disclose the method of claim 15, further comprising triggering dynamic verification depending on a context in which the next instruction is submitted for execution, wherein the context is a level of security clearance associated with the computer, a user of the computer, or a program of which the next instruction is a part [*it is inherent that a computer OS has a method for access control based on the user of the system in that the program will not load if the user does not have permission to run it*].

Regarding claim 26:

Das and Nachenberg '013 disclose the method of claim 2, but does not disclose wherein the response comprises termination of a software entity with which the current memory block is associated. Examiner takes official notice that suspending or cancelling a current software's execution when a virus has been detected was well

known in the art at the time of invention. All the claimed elements were known in the prior art and it would have been obvious to one skilled in the art to combine the elements as claimed by known methods with no change in their respective functions, and the combination would have yielded predictable results to one of ordinary skill in the art at the time of invention.

Regarding claim 27:

Das and Nachenberg '013 disclose the method of claim 2, but does not disclose wherein the response comprises suspension of execution of a software entity with which the current memory block is associated. Examiner takes official notice that suspending or cancelling a current software's execution when a virus has been detected was well known in the art at the time of invention. All the claimed elements were known in the prior art and it would have been obvious to one skilled in the art to combine the elements as claimed by known methods with no change in their respective functions, and the combination would have yielded predictable results to one of ordinary skill in the art at the time of invention.

Regarding claim 30:

Das and Nachenberg '013 disclose the method of claim 2, wherein: the computer includes a virtual machine running on an underlying hardware platform via an intermediate software layer; and the response includes checkpointing the state of the virtual machine [*013, column 3 line 6-18, the CPU emulator is a virtual machine that tricks the virus into thinking it is being run on the CPU*].

Claims 29, 33, 46, 47 and 59 are rejected under 35 U.S.C. 103(a) as being unpatentable over Das and Nachenberg '013 as applied to claims 2 and 35 above, and further in view of SimOS.

Das and Nachenberg '013 disclose the method of claim 2, wherein:

the computer includes a virtual machine running in a direct execution mode on an underlying hardware platform via an intermediate software layer [*013, column 3 lines 6-16, the CPU emulator is a virtual machine that tricks the virus into thinking it is being run on the CPU*], but does not disclose the response comprises a switching of an execution mode of the virtual machine from the direct execution mode to a binary translation mode. SimOS discloses the ability to switch between direct execution and binary translation modes [*page 40, Switching simulators and sampling*]. It would have been obvious to one or ordinary skill in the art at the time of invention to modify the method of Das and Nachenberg '013 to allow switching of modes in order to dynamically generate code supports on-the-fly changes of the simulator's level of detail [*page 39, column 2 paragraph 3*].

Claims 9-11, 16, 18, 22, 24, 25, 28, 31, 32, 42-44, 57 and 58 are rejected under 35 U.S.C. 103(a) as being unpatentable over Das and Nachenberg '013 as applied to claims 2, 8 and 15 above, and further in view of Nachenberg '510.

Regarding claims 9, 42 and 57:

Das and Nachenberg '013 disclose the method of claim 8, but they do not disclose wherein the structure comprises a group of hardware attribute indicators, and wherein the indicating in the structure whether the plurality of memory blocks is

validated comprises setting one of the hardware attribute indicators, the one hardware attribute indicator corresponding to the memory block. Nachenberg '015 discloses a group of hardware attribute indicators, and wherein the indicating in the structure whether the plurality of memory blocks is validated comprises setting one of the hardware attribute indicators, the one hardware attribute indicator corresponding to the memory block [*510, column 3 lines 46-54, register means includes hardware, software, and/or firmware registers, stacks, flags, automata, indication bits, etc.*]. All the claimed elements were known in the prior art and it would have been obvious to one skilled in the art to combine the elements as claimed by known methods with no change in their respective functions, and the combination would have yielded predictable results to one of ordinary skill in the art at the time of invention.

Regarding claims 10 and 43:

013, 510 and Das disclose the method as in claim 9, in which the hardware attribute indicators are execute and write permission attributes associated with an entry in a translation lookaside buffer [Nachenberg '*510, column 3 lines 46-54, register means includes hardware, software, and/or firmware registers, stacks, flags, automata, indication bits, etc.*].

Regarding claims 11, 44 and 58:

013, 510 and Das disclose the method of claim 8, wherein the structure comprises a software data structure, and wherein the indicating in the structure whether the plurality of memory blocks is validated comprises making a corresponding entry in the software data structure [Nachenberg '*510, column 3 lines 46-54, register means*

includes hardware, software, and/or firmware registers, stacks, flags, automata, indication bits, etc.; a stack is a structure].

Regarding claim 16:

013, 510 and Das disclose the method of claim 15, wherein the triggering event is writing of at least one new unit of code or data to any physical component within the computer [Nachenberg '*510, column 3 lines 35-36, the file changed thus causing a new unit of data*].

Regarding claim 18:

013, 510 and Das disclose the method of claim 15, wherein the triggering event is an attempted execution of any instruction located on any unverified memory block of newly installed software [Nachenberg '*510, column 3 lines 26-40, the newly installed software would be scanned on the fact that it is being examined for the first time*].

Regarding claim 22:

013, 510 and Das disclose the method of claim 2, wherein the identifying of the next instruction is performed for only a sample of the series of instructions [*510, figure 1 shows the file is divided into a plurality of sectors and only sectors 1,2,3 and J are scanned*].

Regarding claim 23:

Das and Nachenberg '013 disclose the method of claim 22, wherein the sample is a time- sampled sub-set of the series of instructions [*it is inherent that the virus scanner relies upon some period of time elapsing between samples*].

Regarding claim 24:

013, 510 and Das disclose the method of claim 22, wherein the sample is a sequentially sampled sub-set of the series of instructions [Nachenberg '*510, figure 1 shows the sectors 1, 2 and 3; it is inherent that these sectors would be sequentially sampled*].

Regarding claim 25:

013, 510 and Das disclose the method of claim 22, wherein the sample is a sub-set of the series of instructions sampled spatially, the sampling being over a range of memory block identifiers [Nachenberg '*510, figure 1 shows the sectors 1, 2, 3 are a set size and spatially sampled*].

Regarding claim 28:

013, 510 and Das disclose the method of claim 2, wherein the response comprises a message posted to a user, system administrator, or other predetermined recipient [Nachenberg '*510, column 4 lines 48-63, sends a message to the user to via the interface*].

Regarding claim 31:

013, 510 and Das disclose the method of claim 2, wherein the response is a first possible response, the method further comprising: associating the first possible response with the memory block; associating a second possible response with a different memory block; upon detection of failure of the next instruction to satisfy the validation condition, identifying which one of the possible responses is associated with the memory block, and generating the one possible response associated with the memory block in which the next instruction is located [Nachenberg '*510, column 4 lines,*

determination is sent to the user, it is inherent that the determination will be different for different memory].

Regarding claim 32:

013, 510 and Das disclose the method of claim 2, further comprising: associating reference values from the set of reference values with respective programs such that each association signifies that the reference value corresponds to a memory block storing instructions for one of the programs; and tracking which of the respective programs is being executed and the association between the matching reference value and the corresponding one of the programs [Nachenberg '*510, column 1 lines 27-36, a hash value is associated with a program/file*].

Conclusion

Any inquiry concerning this communication or earlier communications from the examiner should be directed to JAMES TURCHEN whose telephone number is (571)270-1378. The examiner can normally be reached on MTWRF 7:30-5:00.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Edan Orgad can be reached on (571)272-7884. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

JRT

/Michael J Simitoski/
Primary Examiner, Art Unit 2439